Application Report
SPRABQ8–July 2013

Scalar (V/f) Control of 3-Phase Induction Motors

Bilal Akin and Nishant Garg

ABSTRACT

This application report presents a solution to control an AC induction motor using the TMS320F2803x microcontrollers. TMS320F2803x devices are part of the family of C2000™ microcontrollers that enable cost-effective design of intelligent controllers for 3-phase motors by reducing the system components and increase efficiency. In this system, the scalar control (V/Hz) of induction motor experiments with and explores the use of speed control. The user can quickly start evaluating the performance of the V/Hz system with this method.

This application note covers the following:

- A theoretical background on scalar motor control principle.
- Incremental build levels based on modular software blocks.
- Experimental results

Contents

List of Figures

List of Tables

1 Introduction

The motor control industry is a strong, aggressive sector. To remain competitive, new products must address several design constraints including cost reduction, power consumption reduction, power factor correction, and reduced EMI radiation. In order to meet these challenges, advanced control algorithms are necessary. Embedded control technology allows both a high level of performance and system cost reduction to be achieved. According to market analysis, the majority of industrial motor applications use AC induction motors. The reasons for this are higher robustness, higher reliability, lower prices and higher efficiency (up to 80%) on comparison with other motor types. However, the use of induction motors is challenging because of its complex mathematical model, its non linear behavior during saturation and the electrical parameter oscillation that depends on the physical influence of the temperature. These factors make the control of induction motor complex and call for use of a high performance control algorithms such as "vector control" and a powerful microcontroller to execute this algorithm.

Scalar control is the term used to describe a simpler form of motor control, using non-vector controlled drive schemes. An ACI motor can be led to steady state by simple voltage fed, current controlled, or speed controlled schemes. The scalar variable can be manipulated after obtaining its value either by direct measurement or calculation, and can be used in both open loop and closed loop feedback formats. Although its transient behavior is not ideal, a scalar system leads to a satisfactory steady state response.

2 Induction Motors

Induction motors derive their name from the way the rotor magnetic field is created. The rotating stator magnetic field induces currents in the short circuited rotor. These currents produce the rotor magnetic field, which interacts with the stator magnetic field, and produces torque, which is the useful mechanical output of the machine.

2.1

The three-phase squirrel cage AC induction motor is the most widely used motor. The bars forming the conductors along the rotor axis are connected by a thick metal ring at the ends, resulting in a short circuit as shown in Figure 1. The sinusoidal stator phase currents fed in the stator coils create a magnetic field rotating at the speed of the stator frequency (ω_s). The changing field induces a current in the cage conductors, which results in the creation of a second magnetic field around the rotor wires. As a consequence of the forces created by the interaction of these two fields, the rotor experiences a torque and starts rotating in the direction of the stator field.

As the rotor begins to speed up and approach the synchronous speed of the stator magnetic field, the relative speed between the rotor and the stator flux decreases, decreasing the induced voltage in the stator and reducing the energy converted to torque. This causes the torque production to drop off, and the motor will reach a steady state at a point where the load torque is matched with the motor torque. This point is an equilibrium reached depending on the instantaneous loading of the motor. In brief:

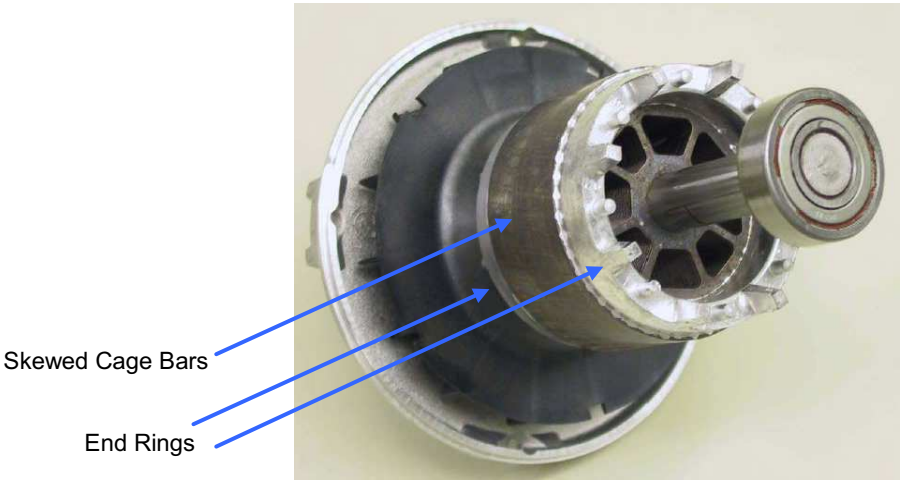

Figure 1. Induction Motor Rotor

- Owing to the fact that the induction mechanism needs a relative difference between the motor speed and the stator flux speed, the induction motor rotates at a frequency near, but less than that of the synchronous speed.
- This slip must be present, even when operating in a field-oriented control regime.
- The rotor in an induction motor is not externally excited. This means that there is no need for slip rings and brushes. This makes the induction motor robust, inexpensive and need less maintenance.
- Torque production is governed by the angle formed between the rotor and the stator magnetic fluxes.

In Figure 2, the rotor speed is denoted by Ω. Stator and rotor frequencies are linked by a parameter called the slip *s*, expressed in per unit as $s = (\omega_s - \omega_r) / \omega_s$.

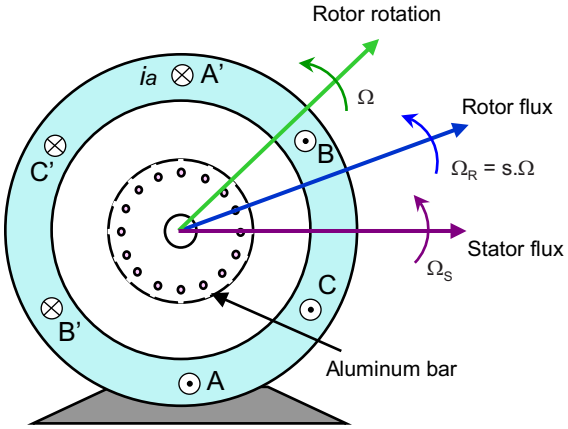

Figure 2. Squirrel Cage Rotor AC Induction Motor Cutaway View

$$\text{– stator rotating field speed } (rad/s): \Omega_S = \frac{\omega_S}{p} \bullet \begin{cases} \omega_S : \text{AC supply freq } (rad/s) \\ p : \text{stator poles pairs numbers} \end{cases}$$

$$\text{– rotor rotating speed } (rad/s): \Omega = (1-s)\Omega_S = (1-s)\frac{\omega_S}{p}$$

where, s is called the “slip”: it represents the difference between the synchronous frequency and the actual motor rotating speed.

3 Scalar Control

3.1 Technical Background

In the V/Hz control, the speed of induction motor is controlled by the adjustable magnitude of stator voltages and frequency in such a way that the air gap flux is always maintained at the desired value at the steady-state. Sometimes this scheme is called the scalar control because it focuses only on the steady-state dynamic. It can explain how this technique works by looking at the simplified version of the steady-state equivalent circuit as seen in Figure 3. According to in this figure, the stator resistance (Rs) is assumed to be zero and the stator leakage inductance (Lls) is embedded into the (referred to stator) rotor leakage inductance (Llr) and the magnetizing inductance, which is representing the amount of air gap flux, is moved in front of the total leakage inductance (Ll = Lls + Llr). As a result, the magnetizing current that generates the air gap flux can be approximately the stator voltage to frequency ratio. Its phasor equation (for steady-state analysis) can be seen as:

$$\tilde{I}_m \cong \frac{\tilde{V}_s}{j\omega L_m} \tag{1}$$

If the induction motor is operating in the linear magnetic region, the Lm is constant. Then, Equation 1 can be shown in terms of magnitude as:

$$I_m = \frac{\Lambda_m}{L_m} \cong \frac{V_s}{(2\pi f)L_m} \tag{2}$$

where, V and Λ are their magnitude of stator voltage and stator flux, and $\tilde{V}$ is the phasor representation, respectively.

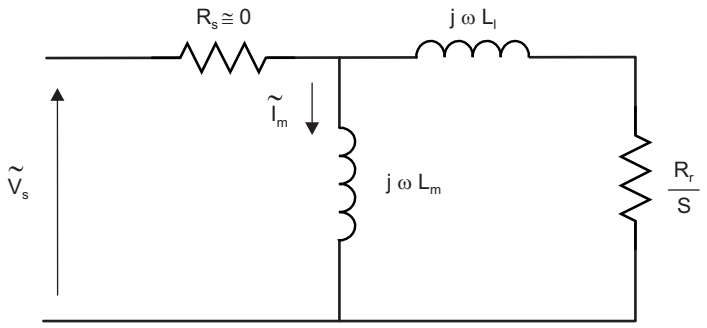

Figure 3. Simplified Steady-State Equivalent Circuit of Induction Motor

From the last equation, it follows that if the ratio V/f remains constant for any change in f, then flux remains constant and the torque becomes independent of the supply frequency. In order to keep Λ_M constant, the ratio of V_s / f would also be constant at the different speed. As the speed increases, the stator voltages must, therefore, be proportionally increased in order to keep the constant ratio of Vs/f. However, the frequency (or synchronous speed) is not the real speed because of a slip as a function of the motor load. At no-load torque, the slip is very small, and the speed is nearly the synchronous speed. Thus, the simple open-loop Vs/f (or V/Hz) system cannot precisely control the speed with a presence of load torque. The slip compensation can be simply added in the system with the speed measurement. The closed-loop V/Hz system with a speed sensor can be shown in Figure 4.

In practice, the stator voltage to frequency ratio is usually based on the rated values of these variables. The typical V/Hz profile can be shown in Figure 5. Basically, there are three speed ranges in the V/Hz profile as follows:

- At 0-fc Hz, a voltage is required, so the voltage drop across the stator resistance cannot be neglected and must be compensated for by increasing the Vs. So, the V/Hz profile is not linear. The cutoff frequency (fc) and the suitable stator voltages may be analytically computed from the steady-state equivalent circuit with Rs ≠ 0.
- At fc-frated Hz, it follows the constant V/Hz relationship. The slope actually represents the air gap flux quantity as seen in Equation 2.
- At higher f_{rated} Hz, the constant Vs/f ratio cannot be satisfied because the stator voltages would be limited at the rated value in order to avoid insulation breakdown at stator windings. Therefore, the resulting air gap flux would be reduced, and this will unavoidably cause the decreasing developed torque correspondingly. This region is usually so called “fieldweakening region”. To avoid this, constant V/Hz principle is also violated at such frequencies.

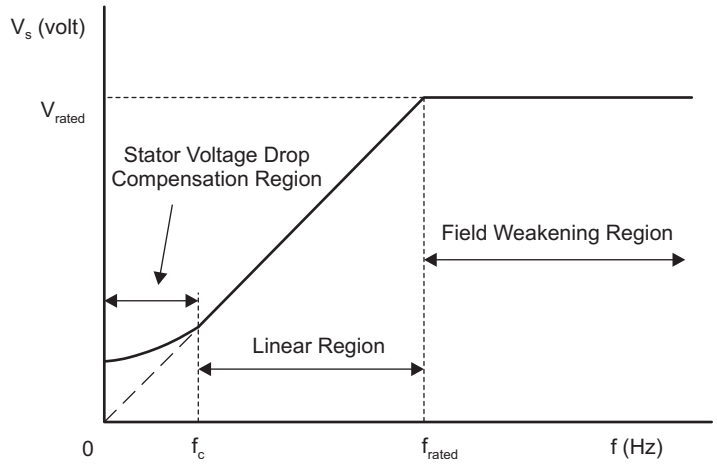

Figure 4. Stator Voltage Versus Frequency Profile Under V/Hz Control

Since the stator flux is constantly maintained (independent of the change in supply frequency), the torque developed depends only on the slip speed. This is shown in Figure 5. By regulating the slip speed, the torque and speed of an AC induction motor can be controlled with the constant V/Hz principle.

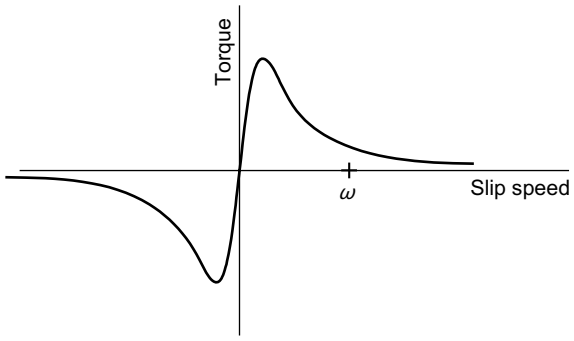

Figure 5. Torque Versus Slip Speed of an Induction Motor With Constant Stator Flux

Both open and closed-loop control of the speed of an AC induction motor can be implemented based on the constant V/Hz principle. Open-loop speed control is used when accuracy in speed response is not a concern such as in HVAC (heating, ventilation and air conditioning), fan or blower applications. In this case, the supply frequency is determined based on the desired speed and the assumption that the motor will roughly follow its synchronous speed. The error in speed resulted from slip of the motor is considered acceptable.

In this implementation, the profile in Figure 4 is modified by imposing a lower limit on frequency, which is shown in Figure 6. This approach is acceptable to applications such as fan and blower drives where the speed response at low end is not critical. Since the rated voltage, which is also the maximum voltage, is applied to the motor at rated frequency, only the rated minimum and maximum frequency information is needed to implement the profile.

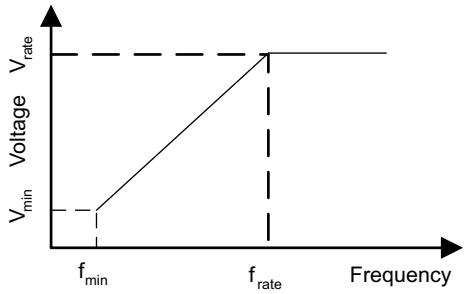

Figure 6. Modified V/Hz Profile

4 Benefits of 32-Bit C2000 Controllers for Digital Motor Control (DMC)

The C2000 family of devices possess the desired computation power to execute complex control algorithms along with the right mix of peripherals to interface with the various components of the DMC hardware like the analog-to-digital converter (ADC), enhanced pulse width modulator (ePWM), quadrature encoder pulse (QEP), enhanced capture (eCAP), and so forth. These peripherals have all the necessary hooks for implementing systems, which meet safety requirements, like the trip zones for PWMs and comparators. Along with this, the C2000 ecosystem of software (libraries and application software) and hardware (application kits) help in reducing the time and effort needed to develop a Digital Motor Control solution. The DMC Library provides configurable blocks that can be reused to implement new control strategies. The IQMath Library enables easy migration from floating-point algorithms to fixed point, thus, accelerating the development cycle.

It is easy and quick to implement complex control algorithms (sensored and sensorless) for motor control with the C2000 family of devices. The use of C2000 devices and advanced control schemes provides the following system improvements:

- Favors system cost reduction by an efficient control in all speed ranges implying right dimensioning of power device circuits
- Using advanced control algorithms, it is possible to reduce torque ripple, resulting in lower vibration and longer life time of the motor.
- Advanced control algorithms reduce harmonics generated by the inverter, thus, reducing filter cost.
- Use of sensorless algorithms eliminates the need for speed or position sensor.
- Decreases the number of look-up tables, which reduces the amount of memory required
- The real-time generation of smooth near-optimal reference profiles and move trajectories results in better-performance
- Generation of high resolution PWM's is possible with the use of ePWM peripheral for controlling the power switching inverters
- Provides single chip control system

For advanced controls, C2000 controllers can also perform the following:

- Enables control of multi-variable and complex systems using modern intelligent methods such as neural networks and fuzzy logic
- Performs adaptive control. C2000 controllers have the speed capabilities to concurrently monitor the system and control it. A dynamic control algorithm adapts itself in real time to variations in system behavior.
- Performs parameter identification for sensorless control algorithms, self commissioning, online parameter estimation update
- Performs advanced torque ripple and acoustic noise reduction

- Provides diagnostic monitoring with spectrum analysis. By observing the frequency spectrum of mechanical vibrations, failure modes can be predicted in early stages.
- Produces sharp-cut-off notch filters that eliminate narrow-band mechanical resonance. Notch filters remove energy that would otherwise excite resonant modes and possibly make the system unstable.

5 TI Literature and Digital Motor Control (DMC) Library

The Digital Motor Control (DMC) library is composed of functions represented as blocks. These blocks are categorized as transforms and estimators (sliding mode observer, phase voltage calculation, resolver, flux, and speed calculators and estimators), control (signal generation, PID, BEMF commutation, space vector generation), and peripheral drivers (PWM abstraction for multiple topologies and techniques, ADC drivers, and motor sensor interfaces). Each block is a modular software macro and separately documented with source code, use, and technical theory. Check the folders below for the source codes and explanations of macro blocks.

This project can be found at www.ti.com/controlsuite:

- \libs\app_libs\motor_control\math_blocks\v4.0
- \libs\app_libs\motor_control\drivers\f2803x_v2.0

These modules allow users to quickly build, or customize their own systems. The Library supports the three motor types: ACI, BLDC, PMSM. It also comprises both peripheral dependent (software drivers) and target dependent modules.

The DMC Library components have been used by TI to provide system examples. At initialization, all DMC Library variables are defined and inter-connected. At run-time, the macro functions are called in order. Each system is built using an incremental build approach, which allows some sections of the code to be built at a time, so that the developer can verify each section of their application one step at a time. This is critical in real-time control applications where so many different variables can affect the system and many different motor parameters need to be tuned.

NOTE: TI DMC modules are written in form of macros for optimization purposes (for more details, *Optimizing Digital Motor Control (DMC) Libraries* (SPRAAK2)). The macros are defined in the header files. The user can open the respective header file and change the macro definition, if needed. In the macro definitions, there should be a backslash "\" at the end of each line as shown below, which means that the code continues in the next line. Any characters including invisible ones like "space" or "tab" after the backslash will cause compilation error. Therefore, make sure that the backslash is the last character in the line. In terms of code development, the macros are almost identical to C function, and the user can easily convert the macro definition to a C functions.

Example 1. A Typical DMC Macro Definition

```
#define PARK_MACRO(v)                                          \
                                                               \
   v.Ds = _IQmpy(v.Alpha,v.Cosine) + _IQmpy(v.Beta,v.Sine);    \
   v.Qs = _IQmpy(v.Beta,v.Cosine) - _IQmpy(v.Alpha,v.Sine);
```

6 System Overview

This section describes the "C" real-time control framework used to demonstrate the scalar control of induction motors. The "C" framework is designed to run on TMS320F2803x-based controllers on Code Composer Studio™. The framework uses the following modules (refer to pdf documents in the motor control folder explaining the details and theoretical background of each macro).

Table 1. C" Real-Time Control Framework Modules

Macro Names	Explanation
PI	PI Regulator
RC	Ramp Controller (slew rate limiter)
VHZ_PROFILE	Voltage and Hertz Profile
QEP / CAP	QEP and CAP Drives
SPEED_PR	Speed Measurement (based on sensor signal period)
SPEED_FR	Speed Measurement (based on sensor signal frequency)
SVGEN_MF	Space Vector PWM (based on magnitude and frequency)
PWM / PWMDAC	PWM and PWMDAC Drives

In this system, the scalar control (V/Hz) of the induction motor explores the performance of speed control. The user can quickly start evaluating the performance of the V/Hz system.

The HVACI_Scalar project has the following properties:

C Framework		
System Name	**Program Memory Usage 2803x**	**Data Memory Usage 2803x** (1) (2)
HVACI_Scalar	3602	1192

(1) Excluding the stack size
(2) Excluding "IQmath" Look-up Tables

CPU Utilization	
Total Number of Cycles	**459** (1)
CPU Utilization @ 60 Mhz	7.7%
CPU Utilization @ 40 Mhz	11.4%

(1) At 10 kHz ISR frequency. Optional macros excluded (PWMDAC, Datalog).

Table 2. System Features

System Features	
Development /Emulation	Code Composer Studio V4.0(or above) with real-time debugging
Target Controller	TMS320F2803x
PWM Frequency	10kHz PWM (Default), 60kHz PWMDAC
PWM Mode	Symmetrical with a programmable dead band
Interrupts	EPWM1 Time Base CNT_Zero – implements 10 kHz ISR execution rate
Peripherals Used	PWM 1 / 2 / 3 for motor control PWM 6A, 6B, 7A and 7B for DAC outputs QEP1 A,B, I or CAP1

The overall system implementing a 3-ph induction motor V/Hz drive implementation is depicted in Figure 7. The induction motor is driven by the conventional voltage-source inverter. The TMS320F2803x is being used to generate the six pulse width modulation (PWM) signals using a space vector PWM technique, for six power switching devices in the inverter.

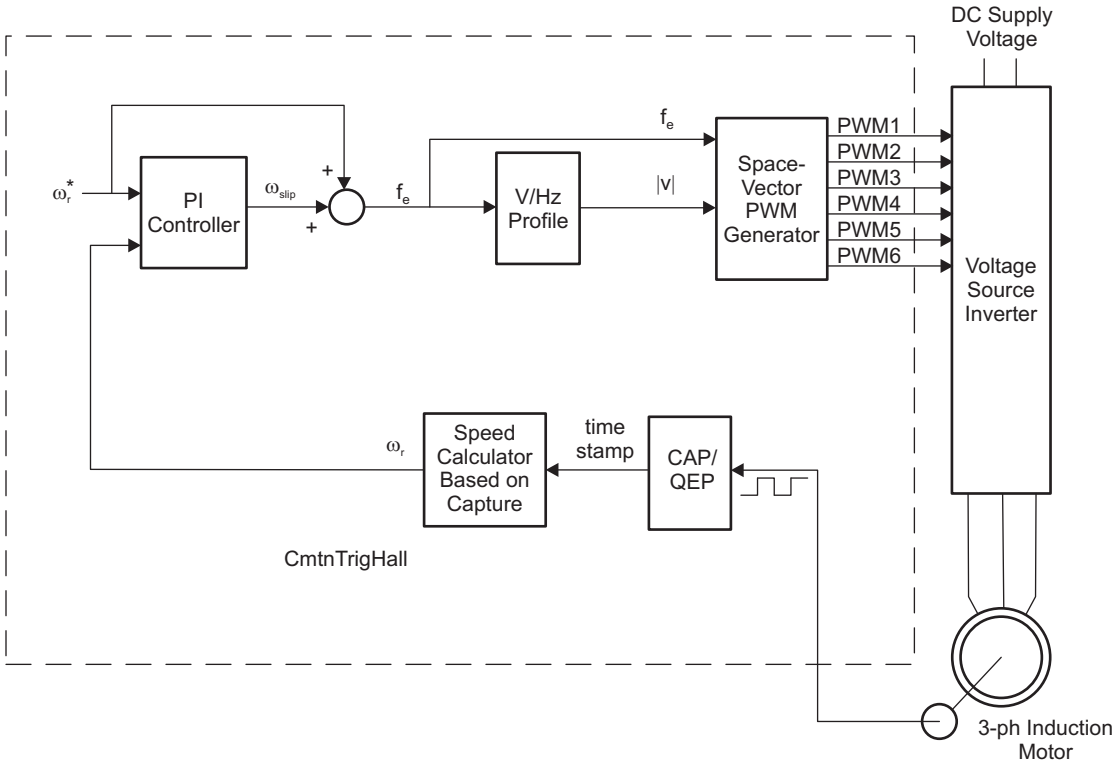

Figure 7. A 3-ph Induction Motor V/Hz Drive Implementation

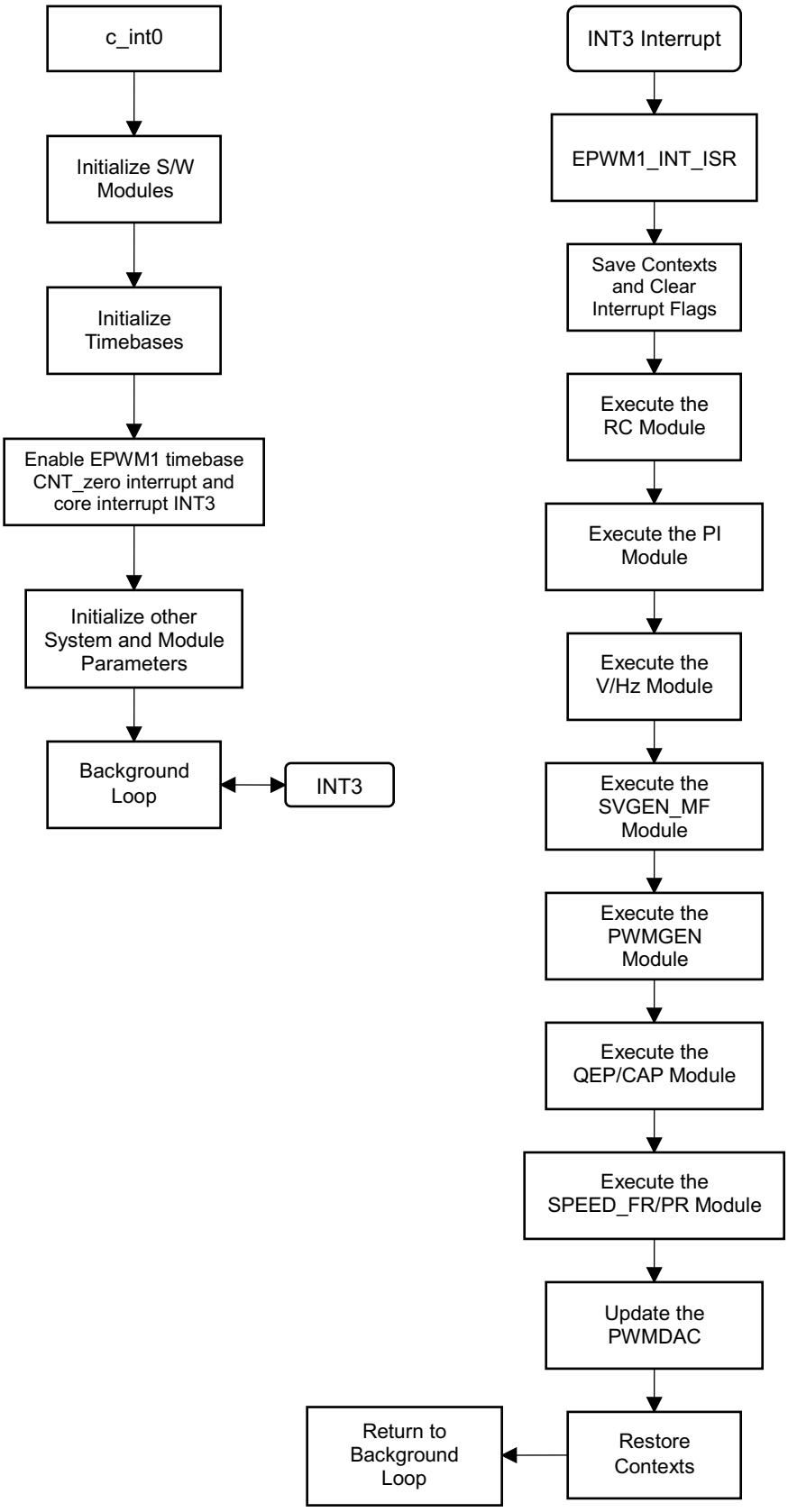

Figure 8. System Software Flowchart

7 Hardware Configuration (HVDMC R1.1 Kit)

For an overview of the kit's hardware and steps on how to setup this kit, see the *HVMotorCtrl+PFC How to Run Guide* located at: www.ti.com/controlsuite and choose the HVMotorKit installation.

7.1 HW Setup Instructions

Some of the hardware setup instructions are listed below for quick reference.

1. Open the lid of the HV kit.
2. Install the Jumpers [Main]-J3, J4 and J5, J9 for 3.3 V, 5 V and 15 V power rails and JTAG reset line.
3. Unpack the DIMM style controlCARD and place it in the connector slot of [Main]-J1. Push down vertically using even pressure from both ends of the card until the clips snap and lock. To remove the card, simply spread open the retaining clip with your thumbs.
4. Connect a USB cable to the connector [M3]-JP1. This enables an isolated JTAG emulation to the C2000 device. [M3]-LD1 should turn on. Make sure [M3]-J5 is not populated. If the included Code Composer Studio is installed, the drivers for the onboard JTAG emulation will automatically be installed. If a windows installation window appears, try to automatically install drivers from those already on your computer. The emulation drivers are found at http://www.ftdichip.com/Drivers/D2XX.htm. The correct driver is the one listed to support the FT2232.
5. If a third party JTAG emulator is used, connect the JTAG header to [M3]-J2 and additionally the [M3]-J5 needs to be populated to put the onboard JTAG chip in reset.
6. Ensure that [M6]-SW1 is in the "Off" position. Connect the 15 V DC power supply to [M6]-JP1.
7. Turn on [M6]-SW1. Now, the [M6]-LD1 should turn on. Notice that the control card LED lights up as well indicating that the control card is receiving power from the board.
8. Note that the motor should be connected to the [M5]-TB3 terminals after you finish with the first incremental build step.
9. Note the DC Bus power should only be applied during incremental build levels when instructed to do so. The two options to get DC Bus power are discussed below:
   - To use DC power supply, set the power supply output to zero and connect [Main]-BS5 and BS6 to DC power supply and ground, respectively.
   - To use AC Mains Power, connect [Main]-BS1 and BS5 to each other using the banana plug cord. Now, connect one end of the AC power cord to [Main]-P1. The other end needs to be connected to the output of a variac. Make sure that the variac output is set to zero and it is connected to the wall supply through an isolator.

For reference, Figure 9 and Figure 10 show the jumper and connectors that need to be connected for this lab.

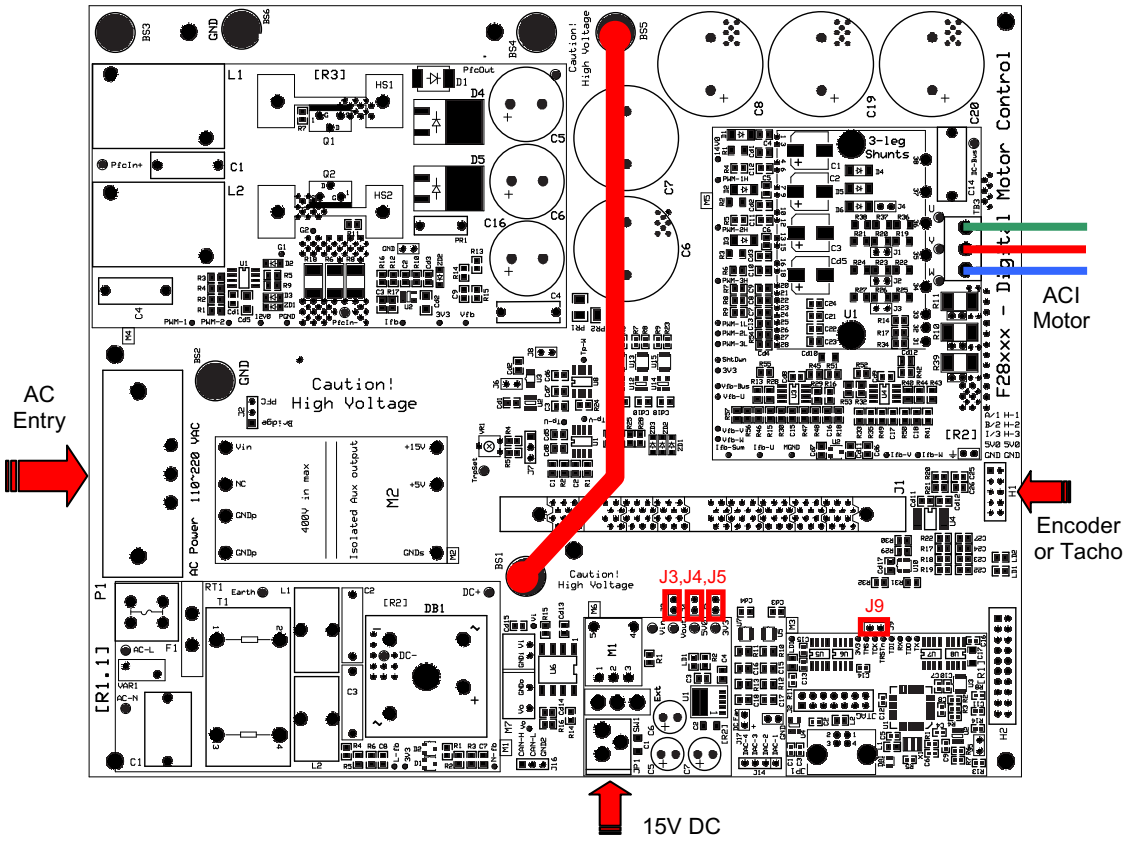

Figure 9. Using AC Power to Generate DC Bus Power

CAUTION

The inverter bus capacitors remain charged for a long time after the high power line supply is switched off or disconnected. Proceed with caution!

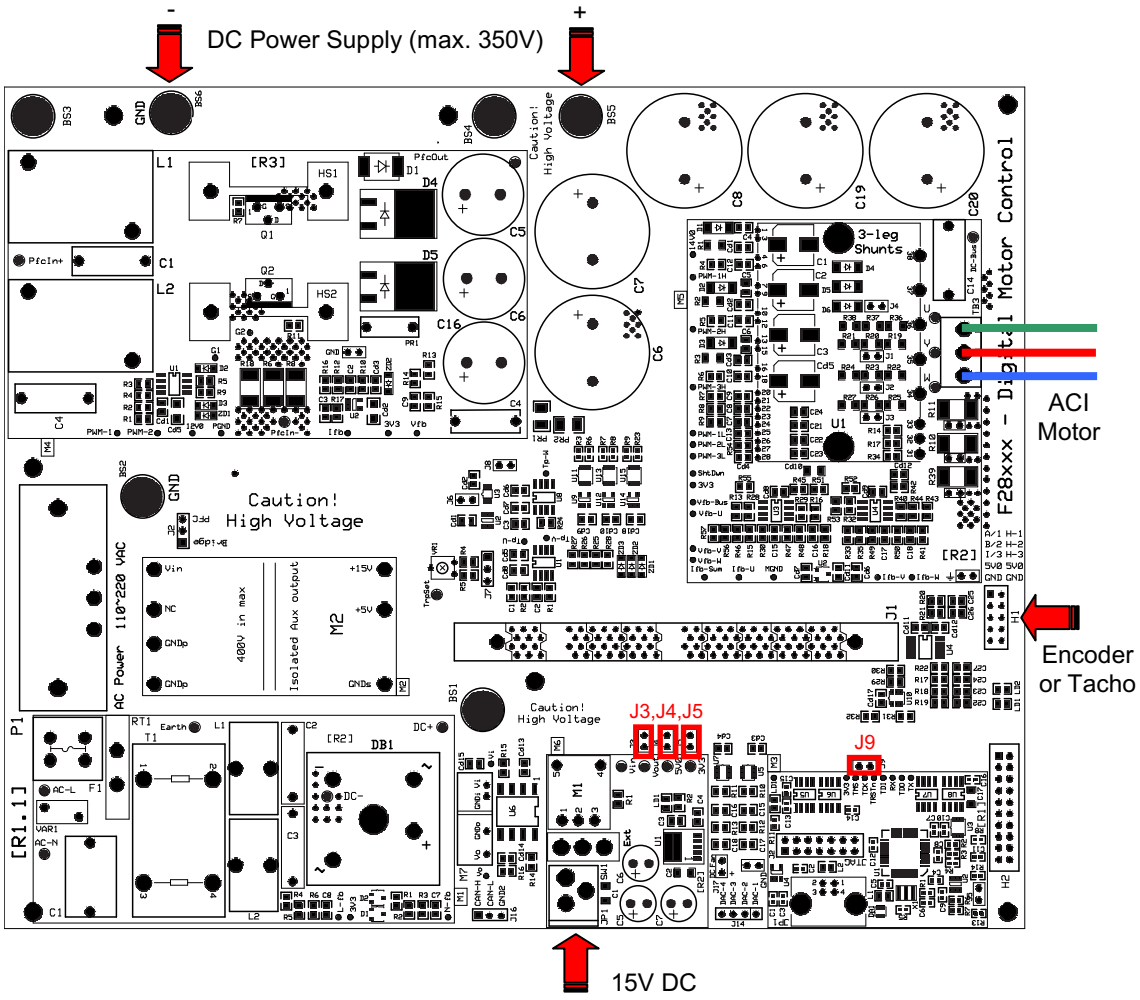

Figure 10. Using External DC Power Supply to Generate DC-Bus for the Inverter

CAUTION

The inverter bus capacitors remain charged for a long time after the high power line supply is switched off or disconnected. Proceed with caution!

7.2 Software Setup Instructions to Run HVACI_Scalar Project

For more information, see the *Software Setup for HVMotorCtrl+PFC Kit Projects* section in the *HVMotorCtrl+PFC Kit How to Run Guide* that can be found at www.ti.com/controlsuite and choose the HVMotorKit installation.

1. Select the HVACI_Scalar as the active project.
2. Select the active build configuration to be set as F2803x_RAM.
3. Verify that the build level is set to 1, and then right click on the project name and select “Rebuild Project”. Once build completes, launch a debug session to load the code into the controller.
4. Open a watch window and add the critical variables as shown in Figure 11 and select the appropriate Q format for them.

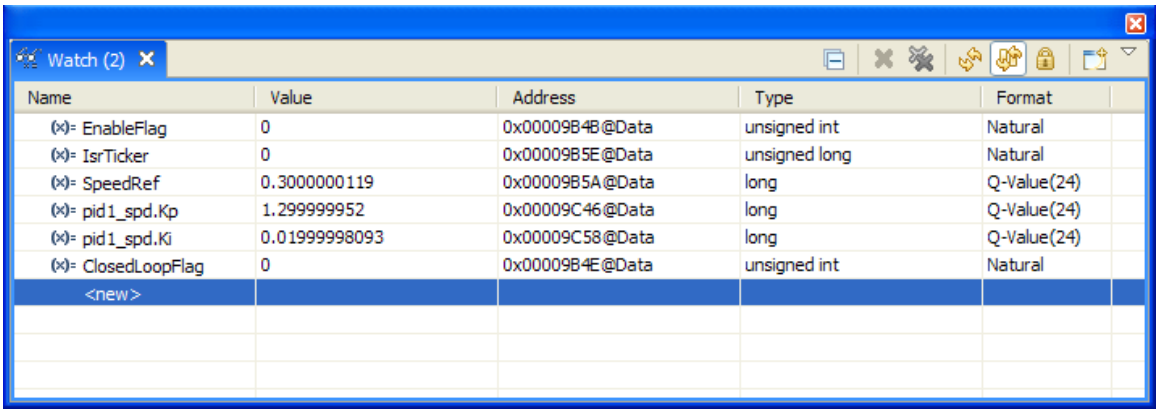

Figure 11. Watch Window Variables

5. Setup the time graph windows by importing Graph1.graphProp and Graph2.graphProp from the following location: www.ti.com/controlsuite - `(developement_kits\HVMotorCtrl+PfcKit_v2.0\HVACI_sensored\).`
6. Go to Tools → Graph → Dual Time, and click the Import button at the bottom of the window.
7. Click on the Continuous Refresh button on the top left corner of the graph tab to enable periodic capture of data from the microcontroller.

8 Incremental System Build

The system is gradually built up so the final system can be confidently operated. Five phases of the incremental system build are designed to verify the major software modules used in the system. Table 3 summarizes the modules testing and using in each incremental system build.

Table 3. Testing Modules in Each Incremental System Build (1)

Software Module	Phase 1	Phase 2	Phase 3
VHZ_PROF_MACRO	√√ (1a)	√	√
SVGEN_MF_MACRO	√√ (1a)	√	√
PWMDAC_MACRO	√√ (1a)	√	√
PWM_MACRO	√√ (1a)	√	√
RC_MACRO		√	√
QEP_MACRO		√√ (2a)	√
SPEED_FR_MACRO		√√ (2a)	√
CAP_MACRO		√√ (2a)	√
SPEED_PR_MACRO		√√ (2a)	v
PI_MACRO (SPD)			√√

(1) The symbol √ means this module is using and the symbol √√ means this module is testing in this phase.

8.1 Level 1 Incremental Build

Keep the motor disconnected at this step. Assuming the load and build steps described in the *HVMotorCtrl+PFC Kit How To Run Guide* completed successfully, this section describes the steps for a "minimum" system check-out, which confirms the operation of the system interrupt, the peripheral and target independent VHZ_PROF_MACRO (V/Hz linearity) and SVGENMF_MACRO (space vector generator) modules.

1. Open HVACI_Scalar-Settings.h and select the level 1 incremental build option by setting the BUILDLEVEL to LEVEL1 (#define BUILDLEVEL LEVEL1).
2. Right click on the project name and click Rebuild Project.

3. Click on the debug button, reset the CPU, restart, enable real-time mode and run, once the build is complete.
4. Set the “EnableFlag” to 1 in the watch window. The variable named “IsrTicker” will now keep on increasing. Confirm this by watching the variable in the watch window. This confirms that the system interrupt is working properly.

In the software, the key variables to be adjusted are:

SpeedRef (Q24): for changing the rotor speed in per-unit.

8.2 Level 1A (V/Hz and SVGEN_MF Macro Testing)

Before testing this phase, the knowledge of V/Hz profile has to be kept in mind. The base frequency can be specified at 2 times of the rated frequency (60 Hz) in order to run the motor at a speed higher than the rated speed (in the field weakening). Figure 12 shows an example of the V/Hz profile used in this system.

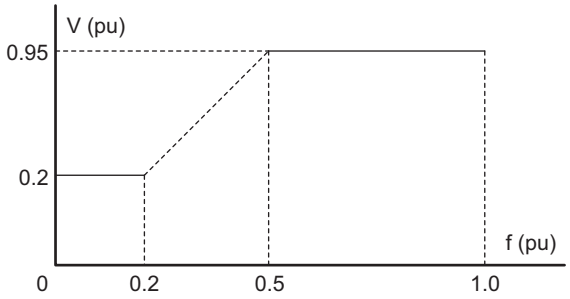

Figure 12. V/Hz Profile Configuration Used in This System

The SpeedRef value is specified to the VHZ_PROF_MACRO module. The VHZ_PROF_MACRO module generates the outputs to the SVGENMF_MACRO module. Three outputs from the SVGENMF_MACRO module are monitored via the graph window as shown in Figure 13 where Ta, Tb, and Tc waveforms are 120° apart from each other. Specifically, Tb lags Ta by 120° and Tc leads Ta by 120°. During the running of this build, the waveforms in the Code Composer Studio graphs should appear as show in Figure 13.

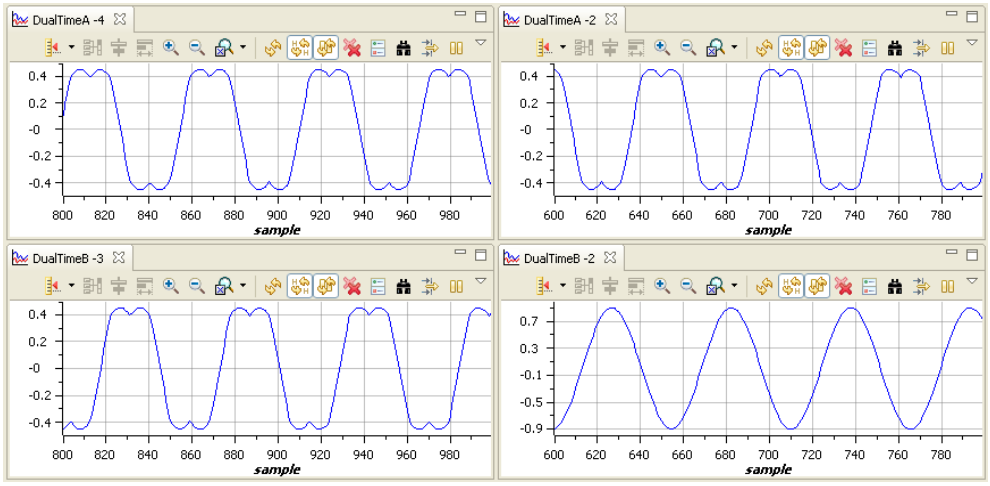

Figure 13. SVGEN Duty Cycle Outputs Ta, Tb, Tc and Ta-Tb

8.3 Level 1B (PWMDAC Macro Testing)

To monitor internal signal values in real-time PWM, DACs are very useful tools. Present on the HV DMC board are PWM DAC’s that use an external low-pass filters to generate the waveforms ([Main]-J14, DAC-1 to 4). A simple first-order low-pass filter RC circuit is used to filter out the high frequency components. The selection of R and C value (or the time constant, τ) is based on the cut-off frequency (f_c), for this type of filter; the relation is as follows:

$$\tau = RC \frac{1}{2\pi f_c}$$

For example, R = 1.8 kΩ and C = 100 nF, it gives f_c = 884.2 Hz. This cut-off frequency has to be below the PWM frequency. Using the formula above, one can customize the low-pass filters used for signal being monitored.

The DAC circuit low-pass filters ([Main]-R10 to13 and [Main]-C15 to18) are shipped with 2.2 kΩ and 220 nF on the board. For more details, see *Using PWM Output as a Digital-to-Analog Converter on a TMS320F280x Digital Signal Controller* (SPRAA88).

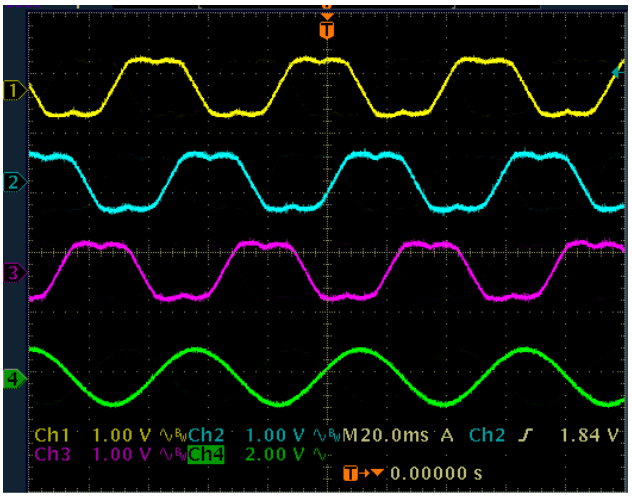

Figure 14. DAC-1-4 Outputs Showing Ta, Tb, Tc and Ta-Tb Waveforms

8.4 Level 1C (PWM_MACRO and INVERTER Testing)

After verifying the SVGENMF_MACRO module in Level 1A, the PWM_MACRO software module and the 3-phase inverter hardware are tested by looking at the low-pass filter outputs. For this purpose, if using the external DC power supply, gradually increase the DC bus voltage and check the Vfb-U, V and W test points using an oscilloscope or if using AC power entry slowly change the variac to generate the DC bus voltage. Once the DC bus voltage is greater than 15 V to 20 V, you will start observing the inverter phase voltage dividers and waveform monitoring filters (Vfb-U, Vfb-V, Vfb-W) enable the generation of the waveform, which ensures that the inverter is working appropriately. Note that the default RC values are optimized for AC motor state observers employing phase voltages.

Once the instructions given in Level1A to 1C are successfully completed and the system is shut down as suggested below, the motor can be connected. Now the system is ready to run open loop.

CAUTION

After verifying this, reduce the DC bus voltage, take the controller out of real-time mode (disable), and reset the processor (for details, see the *HVMotorCtrl+PFC Kit How To Run Guide*). Note that after each test, this step needs to be repeated for safety purposes. Also note that improper shutdown might halt the PWMs at some certain states where high currents can be drawn, therefore, caution needs to be taken while doing these experiments.

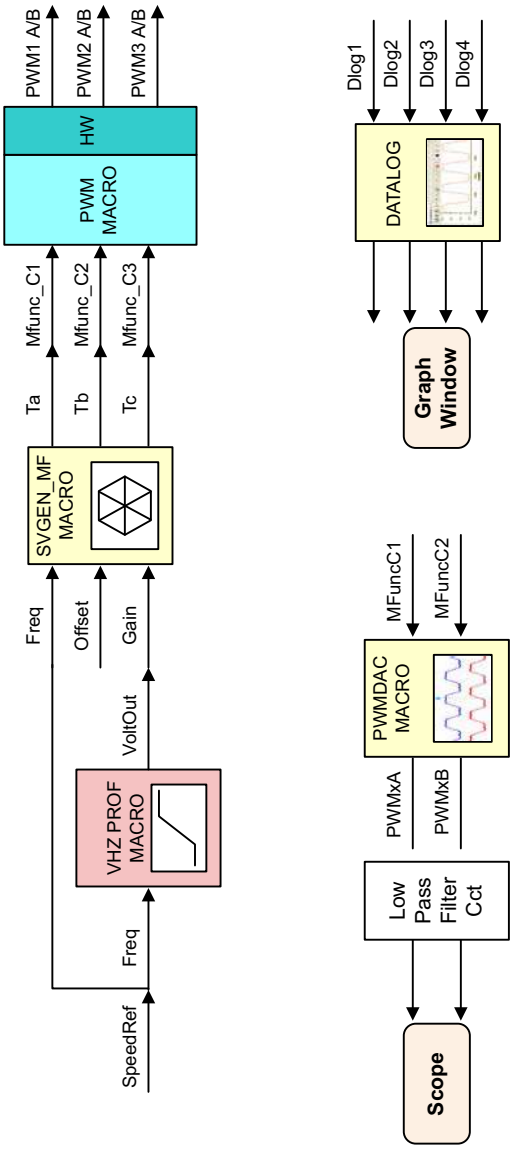

Figure 15. Level 1 - Incremental System Build Block Diagram

Level 1 verifies the target independent modules V/Hz and SVGENMF.

8.5 Level 2 - Incremental Build

Assuming section BUILD 1 is completed successfully, this section verifies the analog-to-digital conversion, Clarke and Park transformations and phase voltage calculations. Now the motor can be connected to the HVDMC board since the PWM signals are successfully proven through level 1 incremental build.

1. Open HVACI_Scalar-Settings.h and select level 2 incremental build option by setting the BUILDLEVEL to LEVEL2 (#define BUILDLEVEL LEVEL2) and save the file.
2. Right Click on the project name and click Rebuild Project.
3. Click on debug button, reset the CPU, restart, enable real-time mode and run, once the build is complete.
4. Set the “EnableFlag” to 1 in the watch window. The variable named “IsrTicker” is incrementally increased as seen in the watch windows to confirm the interrupt working properly.

In the software, the key variables to be adjusted are:

SpeedRef (Q24): for changing the rotor speed in per-unit

8.6 Level 2A – Testing the QEP and SPEED_FR (for incremental encoder)

This section verifies the QEP1 driver and its speed calculation. QEP drive macro determines the rotor position and generates a direction (of rotation) signal from the shaft position encoder pulses. Make sure that the output of the incremental encoder is connected to [Main]-H1 and QEP and SPEED_FR macros are initialized properly in the HVACI_Scalar.c file depending on the features of the speed sensor. The pdf files regarding the details of related macros in the motor control folder are located at: www.ti.com/controlsuite - (\libs\app_libs\motor_control). The steps to verify these two software modules related to the speed measurement can be described as follows:

- Set SpeedRef to 0.4 pu (or another suitable value if the base speed is different).
- Compile/load/run program with real time mode and then increase voltage at variac and dc power supply to get the appropriate DC-bus voltage. Now the motor is running close to reference speed.
- Check the “speed1.Speed” in the watch windows with continuous refresh feature whether or not the measured speed is less than SpeedRef a little bit due to a slip of motor.
- To confirm these modules, try different values of SpeedRef to test the Speed.
- Use oscilloscope to view the electrical angle output, ElecTheta, from QEP_MACRO module and the emulated rotor angle, Out, from RG_MACRO at PWMDAC outputs with external low-pass filters.
- Check that both ElecTheta and Out are of saw-tooth wave shape and have the same period. If the measured angle is in opposite direction, then change the order of motor cables connected to inverter output (TB3 for HVDMC kit).
- Check from Watch Window that qep1.IndexSyncFlag is set back to 0xF0 every time it reset to 0 by hand. Add the variable to the watch window if it is not already in the watch window.

8.7 Level 2B – Testing the CAP and SPEED_PR (for tacho or sprocket)

In this case, the CAP1 input is chosen to detect the edge. If available, make sure that the sensor output is connected to [Main]-H1 and CAP and SPEED_PR macros are initialized properly in the HVACI_Scalar.c file depending on the features of the speed sensor. Typically the capture is used to measure speed when a simple low cost speed sensing system is available. The sensor generates pulses when detecting the teeth of a sprocket or gear and the capture drive provides the instantaneous value of the selected time base (GP Timer) captured on the occurrence of an event. For the details of related macros, see the PDF files located in the motor control folder located at: www.ti.com/controlsuite - `(libs\app_libs\motor_control)`.

The steps to verify these two software modules related to the speed measurement can be described as follows:

- Set SpeedRef to 0.3 pu (or another suitable value if the base speed is different).
- Compile, load, and run the program with real-time mode and then increase the voltage at the variac and dc power supply to get the appropriate DC-bus voltage. Now, the motor is running reference speed.
- Check the “speed2.Speed” in the watch windows with the continuous refresh feature whether or not they should be less than SpeedRef a little bit due to a slip of motor.
- Try different values of SpeedRef to test the speed to confirm these modules.
- Reduce the voltage at the variac and dc power supply to zero, halt the program and stop real-time mode. Now the motor is stopping.

An alternative to verify these two software modules without running the motor can be done by using a function generator. The key steps can be explained as follows:

- Use a function generator to generate the 3.3 V (DC) square-wave with the desired frequency corresponding to the number of teeth in the sprocket and the wanted speed in RPM. Then, connect only the pulse signal and ground wires from the function generator to the HVDMC board. The desired frequency of the square-wave produced by the function generator can be formulated as:

$$f_{square\_wave} = \frac{RPM \times TEETH}{60} Hz$$

  where, RPM is the wanted speed in rpm, and TEETH is the number of teeth in the sprocket.
- Compile, load, and run the program with real-time mode and then increase the voltage at the variac to get the appropriate DC-bus voltage. Now, the motor is running. Note that the SpeedRef could be set to any number.
- Check the speed2.Speed and speed2.SpeedRpm in the watch windows with the continuous refresh feature whether or not they should be corresponding to the wanted speed that is chosen before.
- To confirm these modules, change different frequencies of square-wave produced by function generator with corresponding wanted (known) speed to check the Speed and SpeedRpm.

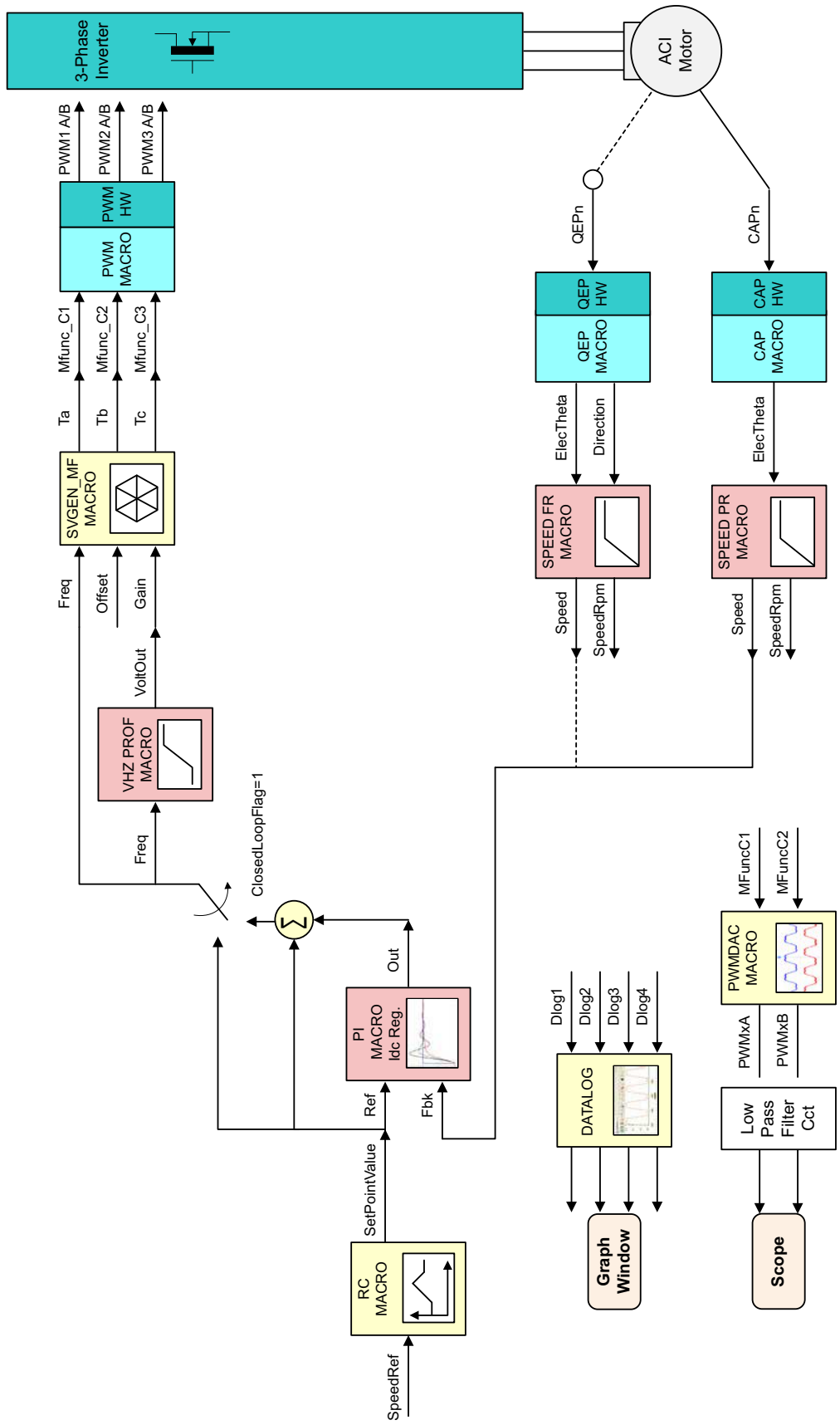

Figure 16. Level 2- Incremental System Build Block Diagram

Level 2 verifies the PWM generation, analog-to-digital conversions, and capture units.

8.8 Level 3 Incremental Build

Assuming the previous section is completed successfully, this section verifies the dq-axis current regulation performed by PI modules and speed measurement modules. To confirm the operation of current regulation, the gains of these two PI controllers are necessarily tuned for proper operation.

1. Open HVACI_Scalar-Settings.h and select the level 3 incremental build option by setting the BUILDLEVEL to LEVEL3 (#define BUILDLEVEL LEVEL3).
2. Right click on the project name and click Rebuild Project.
3. Click on the debug button, reset the CPU, restart, enable real-time mode and run, once the build is complete.
4. Set the “EnableFlag” to 1 in the watch window. The variable named “IsrTicker” is incrementally increased as seen in the watch windows to confirm the interrupt working properly.

In the software, the key variables to be adjusted are summarized below:

- SpeedRef (Q24): for changing the rotor speed in per-unit
- Set ClosedLoopFlag: for switching between open-loop and closed-loop control

In this build, the motor is supplied by AC input voltage and the (AC) motor current is dynamically regulated by using PI module through the park transformation on the motor currents.

The key steps are explained as follows:

- Set SpeedRef to 0.3 pu (or another suitable value if the base speed is different).
- Set ClosedLoop Flag to 0.
- Compile, load, and run the program with real-time mode and then set EnableFlag to 1.
- Gradually increase the voltage at the variac and dc power supply to get an appropriate DC-bus voltage. Now the motor is running with this set speed (0.4 pu).
- Adjust gains Kp, Ki such that the PI output is properly representing the slip.
- Then, set ClosedLoopFlag to 1.
- Try different values of SpeedRef and check speed to confirm these two PI modules. Set ClosedLoopFlag to 0 immediately if the close-loop system is not successfully operated.
- For both PI controllers, the proportional, integral, derivative and integral correction gains may be re-tuned to have the satisfied responses.
- Reduce voltage at variac to zero, halt program and stop real-time mode. Now the motor is stopping.

When running this build, the current waveforms in the Code Composer Studio graphs should appear as shown in Figure 17.

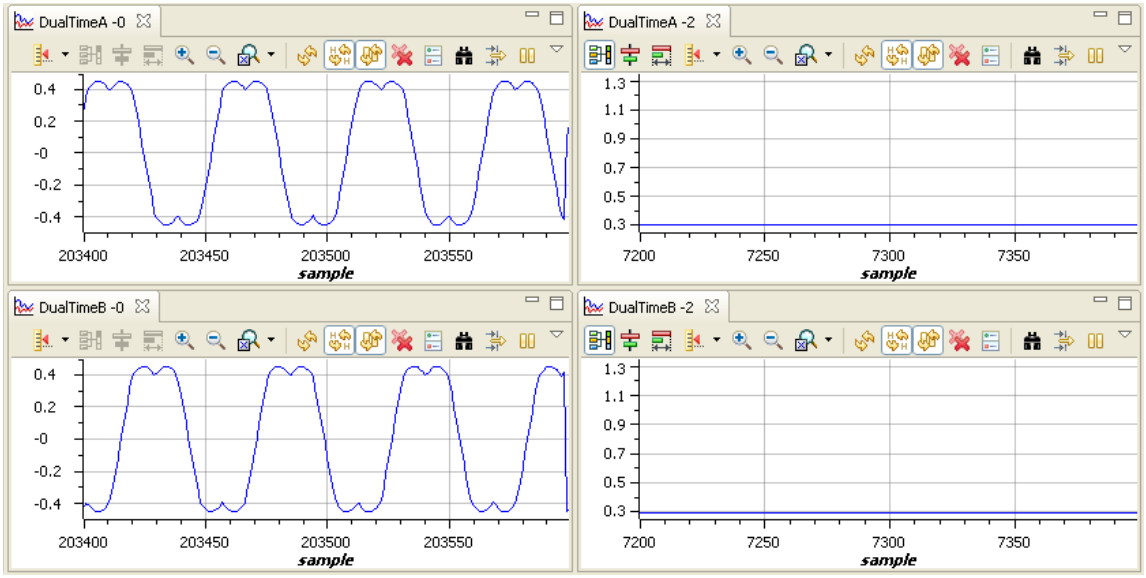

Figure 17. Ta (top-left), Tb (bottom-left), Speed Reference (top-right), Speed Feedback (bottom-right)

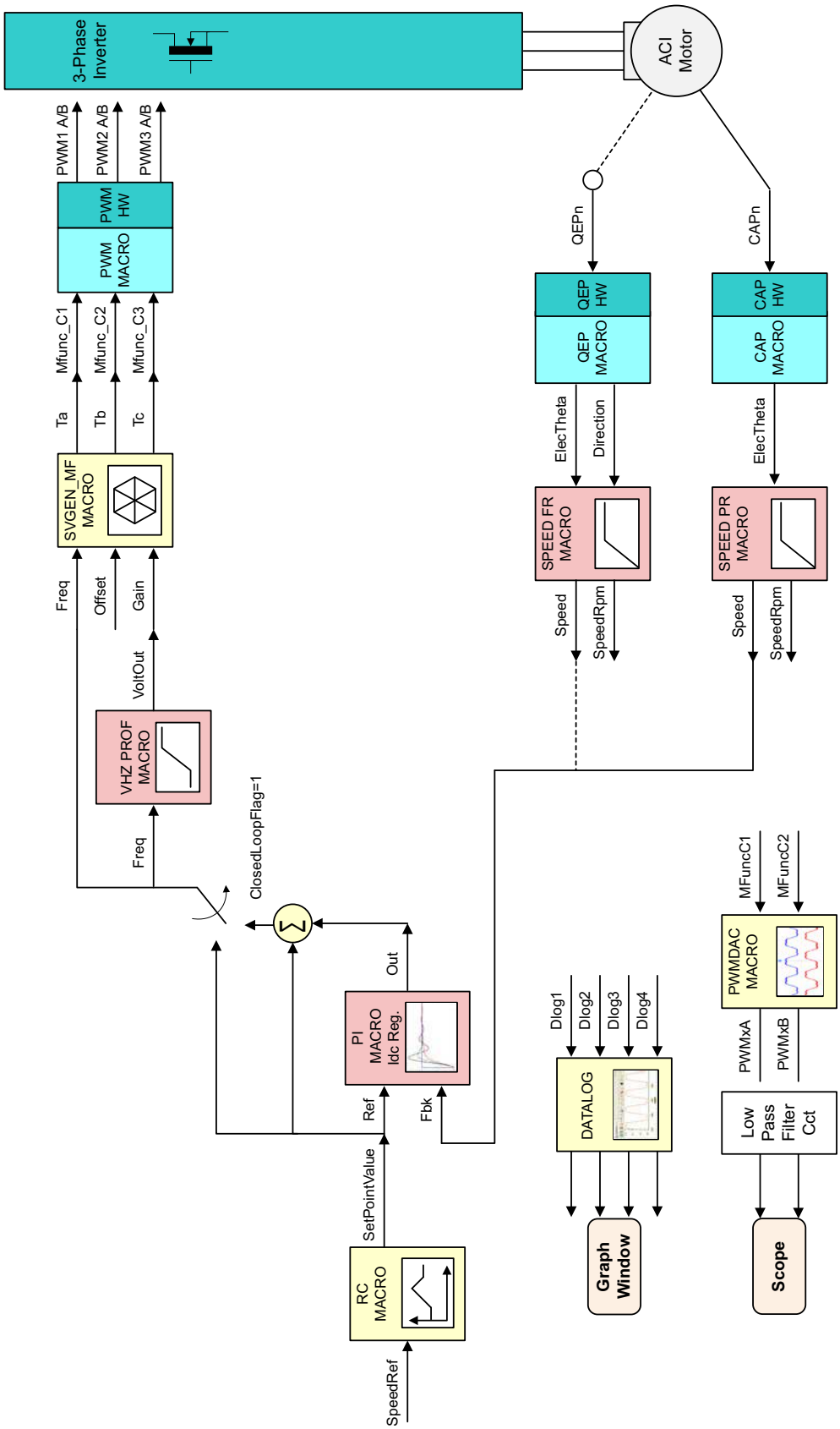

Figure 18. Level 3 - Incremental System Build Block Diagram

Level 3 verifies PI Regulator and Closed Loop functionality.

9 References

- *Optimizing Digital Motor Control (DMC) Libraries* (SPRAAK2)
- *Using PWM Output as a Digital-to-Analog Converter on a TMS320F280x Digital Signal Controller* (SPRAA88)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265